



UvA-DARE (Digital Academic Repository)

Epistemic modelling and protocol dynamics

Wang, Y.

Publication date
2010

[Link to publication](#)

Citation for published version (APA):

Wang, Y. (2010). *Epistemic modelling and protocol dynamics*. [Thesis, fully internal, Universiteit van Amsterdam]. Institute for Logic, Language and Computation.

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

3.1 Introduction

Public announcements are the simplest and best studied communication methods in epistemic logic [vDvdHK07]. In this chapter, we focus on the epistemic protocols that use public announcements as the only communication methods. As we mentioned in the introduction, existing work in the framework of DEL represents epistemic protocols as explicit sets of (finite) sequences of epistemic events (see, e.g., [HY09, vBGHP09, Hos09]). However, as remarked in [PR03], an *explicit* set of sequences of events is an *extensional* notion of *common sense* protocols which are usually specified by a few rules governing the communications. Therefore a high-level, finitely representable and model-independent specification is preferable for epistemic protocols. This motivates us to represent epistemic protocols as (syntactic) programs and thus focus on a subclass of "regular protocols" in this chapter. This restriction allows us to define a dynamic epistemic logic where protocols and their consequences are both expressible in the language. Thus the formal specification and verification are unified in a logical framework.

As we motivated in Chapter 1, complications arise in the verification of epistemic protocols compared to the verification of normal protocols. The correctness of the epistemic protocols heavily relies on the assumptions of the agents' *meta-knowledge* about the protocol itself. In particular, if the intended goal of an epistemic protocol is to establish or prevent knowledge, then knowing that the protocol would fulfil the goal may affect the verification of the protocol. It is reasonable to assume that the protocol, its goal, and the underlying initial assumptions are commonly known by all the agents including possible adversaries. Based on the logic we propose, we can formally address the above subtlety and verify epistemic protocols under the assumption that the intended goal is common knowledge.

Moreover, the formal specification of epistemic protocols calls for a more careful study of the classic problems. For example, recall the Russian Cards Problem (RCP) in Example 1.1.1, where A and B want to safely inform each other of their own cards by using public communications only, with the presence of an adversary E . A

satisfactory protocol to realize this safe information exchange should specify what A and B should do under *any* initial card deals. However, in the previous studies of the Russian Cards Problem (e.g. [vD03, vDvdHvdMR06]), the focus was usually on a particular deal of the cards¹. As we shall see, the framework developed in this chapter can help us design and verify protocols that work on arbitrary initial distributions. The formal discussion also reveals some further subtleties. For example, in the case of $RCP_{3,3,1}$ we can show that correct deterministic protocols that are executable on arbitrary initial distributions of cards do exist, but that they are necessarily biased with respect to single card occurrence in the announcement..

Related work The closest work to ours is [vD02], where, instead of using action models as in [BM04], the author specifies the epistemic events by programs involving atomic epistemic actions such as learning and testing (see also [vDvdHK03c] and [vDvdHK07, Chapter 5] for extensions). Compared to this approach, our focus is on the verification of epistemic protocols, i.e. sequences of epistemic events, which sit at a higher level than the events themselves. This difference is also reflected in the design of the languages. For example, iteration over epistemic events is crucial in our work, while it may not fit in a description of complex epistemic events.

Structure of the chapter An epistemic logic of protocol specification and verification is introduced in Section 3.2, whose model checking problem is shown to be decidable. Section 3.3 formally addresses the specification and verification of epistemic protocols. We show that if the meta-knowledge of the protocol is assumed, then the verification problem should be formalized as model checking a fixed point formula involving iterated announcements. We also define a notion of *universal verification* of epistemic protocols with respect to a model, in which case checking the common knowledge of the correctness of the protocol suffices. To demonstrate the use of our framework, we study the deterministic protocols for the Russian Cards Problem in Section 3.4.

3.2 Preliminaries

In this section we define an *Announcement Protocol Language* L_{AP} for specifying and verifying epistemic protocols with announcements only. Our language is based on test-free PDL but with public announcements as atomic programs. The choice of test-free PDL is based on the observation that each announcement $!\phi$ has an intrinsic guard $?\phi$ which is assumed to be common knowledge in this chapter. For announcements with non-intrinsic tests, e.g. $?K_i(p \wedge q)!\cdot K_i p$, see discussions in Section 3.5 and the next

¹For example, [vDvdHvdMR06] focuses on announcements for the specific situation of the card deal (012.345.6). The authors mentioned that the model checking task of a protocol that provides an announcement for an arbitrary initial state takes much more time, but the protocol itself is not discussed in the paper.

chapter. The formulas of L_{AP} are defined as:

$$\begin{aligned}\phi & ::= \top \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid [\pi]\phi \mid C_G\phi \\ \pi & ::= !\phi \mid \pi_1 \cdot \pi_2 \mid \pi_1 + \pi_2 \mid \pi^*\end{aligned}$$

where p ranges over a set of basic propositions \mathbf{P} and G is a subset of the set of agents \mathbf{I} .

We write $K_i\phi$ for $C_{\{i\}}\phi$ and $C\phi$ for $C_{\mathbf{I}}\phi$. Intuitively, $[\pi]\phi$ expresses “after any possible run of the protocol π , ϕ holds”. We write $\langle\pi\rangle\phi$ for $[\pi]\phi \wedge \langle\pi\rangle\phi$. Thus $\langle\pi\rangle\phi$ says “protocol π is executable and after every possible run ϕ holds”. Moreover, we use $!_i\phi$ as the abbreviation for the announcement $!K_i\phi$. Intuitively, $!_i\phi$ is a public announcement of ϕ by agent i while $!\phi$ models an *external* announcement from the environment (e.g., the role of *Father* in the Muddy Children example). π is called an *n-step* protocol if $\pi = \pi_1 \cdot \pi_2 \cdots \pi_n$ and for $i \leq n : \pi_i$ does not contain operators $*$ and \cdot .

Given an $S5$ model $\mathcal{M} = (S, \mathbf{P}, \mathbf{I}, \sim, V)$, the truth value of a L_{AP} formula ϕ at a state s in \mathcal{M} is defined as:

$\mathcal{M}, s \models p$	\Leftrightarrow	$p \in V(s)$
$\mathcal{M}, s \models \neg\phi$	\Leftrightarrow	$\mathcal{M}, s \not\models \phi$
$\mathcal{M}, s \models \phi \wedge \psi$	\Leftrightarrow	$\mathcal{M}, s \models \phi$ and $\mathcal{M}, s \models \psi$
$\mathcal{M}, s \models C_G\psi$	\Leftrightarrow	$\forall t : s \sim_G t \Rightarrow \mathcal{M}, t \models \psi$
$\mathcal{M}, s \models [\pi]\phi$	\Leftrightarrow	for all $\mathcal{M}', s' : (\mathcal{M}, s) \llbracket \pi \rrbracket (\mathcal{M}', s')$ implies $\mathcal{M}', s' \models \phi$

where $\sim_G = (\bigcup_{i \in G} \sim_i)^*$ and π are the epistemic programs functioning as *model changers*:

$(\mathcal{M}, s) \llbracket !\psi \rrbracket (\mathcal{M}', s)$	\Leftrightarrow	$(\mathcal{M}', s) = (\mathcal{M} _{\psi}, s)$
$(\mathcal{M}, s) \llbracket \pi_1 \cdot \pi_2 \rrbracket (\mathcal{M}', s)$	\Leftrightarrow	$(\mathcal{M}, s) \llbracket \pi_1 \rrbracket \circ \llbracket \pi_2 \rrbracket (\mathcal{M}', s)$
$(\mathcal{M}, s) \llbracket \pi_1 + \pi_2 \rrbracket (\mathcal{M}', s)$	\Leftrightarrow	$(\mathcal{M}, s) \llbracket \pi_1 \rrbracket \cup \llbracket \pi_2 \rrbracket (\mathcal{M}', s)$
$(\mathcal{M}, s) \llbracket \pi^* \rrbracket (\mathcal{M}', s)$	\Leftrightarrow	$(\mathcal{M}, s) \llbracket \pi \rrbracket^* (\mathcal{M}', s)$

where $\mathcal{M}|_{\psi}$ is the restriction of \mathcal{M} to the states where ψ holds (see Section 2.3.3); \circ , \cup and $*$ express the usual composition, union and reflexive transitive closure on relations respectively. Viewing π as a regular expression over $\{!\phi \mid \phi \text{ is an } L_{AP} \text{ formula}\}$, we have:

$$(\mathcal{M}, s) \llbracket \pi \rrbracket (\mathcal{M}', s) \iff \text{there is a sequence } w \in \mathcal{L}(\pi) \text{ such that } (\mathcal{M}, s) \llbracket w \rrbracket (\mathcal{M}', s)$$

A *run* of π on a model (\mathcal{M}, s) is a sequence w of announcements such that $w \in \mathcal{L}(\pi)$ and $(\mathcal{M}, s) \llbracket w \rrbracket (\mathcal{M}', s)$ for some (\mathcal{M}', s) . For each run w on (\mathcal{M}, s) there is a unique path of pointed models $(\mathcal{M}, s), (\mathcal{M}_1, s), \dots, (\mathcal{M}_n, s)$ such that $\mathcal{M}_n = \mathcal{M}'$, which realizes w .

Iteration is important in specifying epistemic protocols with **while-do** loops (see for example, [vDvdHK07, pp.13] and [DW07]). We will also show, in the next section, that having the Kleene star in the specification language is crucial for verifying epistemic protocols. However, [MM05] showed that the satisfiability problem of a logic containing both iterated announcement $((!\phi)^*)$ and common knowledge operators is undecidable, even on finite models. Fortunately, the model checking problem of L_{AP} on finite models is decidable:

3.2.1. PROPOSITION. *Model checking L_{AP} on finite Kripke models is decidable.*

PROOF The idea of the proof is based on the observation that the basic epistemic programs $!\phi$ of L_{AP} are *eliminative* in nature, which means that the transformed model is only getting smaller after a run of an epistemic program. The aim is then to reduce model checking ϕ on (\mathcal{M}, s) to the standard PDL-style model checking of ϕ on a larger finite model \mathcal{N} , where programs π in ϕ are taken as labels of relations in \mathcal{N} . The state space of \mathcal{N} is the set of all the possible pointed sub-models (\mathcal{M}', s') of \mathcal{M} . We define $(\mathcal{M}', s') \sim_i^{\mathcal{N}} (\mathcal{M}'', s'') \iff (\mathcal{M}' = \mathcal{M}'' \text{ and } s' \sim_i s'' \text{ in } \mathcal{M}')$. We let the valuation $V_{\mathcal{N}}((\mathcal{M}', s')) = V_{\mathcal{M}}(s')$. Now we are ready to compute all the corresponding relations of π in \mathcal{N} by usual treatments in PDL model checking algorithms for operators $\cdot, +$ and $*$ and the following operation to deal with $!\phi'$: $\mathcal{M}', s' \rightarrow_{!\phi'} \mathcal{M}'', s''$ iff $\mathcal{M}'' = \mathcal{M}'|_{\phi'}$ and $s' = s''$. To compute $\mathcal{M}'|_{\phi'}$ we need to call the model checking algorithm again but since ϕ' (a subformula of ϕ) is strictly simpler than ϕ , we will finally arrive at a situation that can be handled by the PDL model checking algorithm. \aleph

3.3 Announcement Protocol and Verification

In this section, we specify the announcement protocols and address their verification problem formally.

To verify an epistemic protocol, it is important to specify the assumptions about the initial setting in which the protocol is to be executed. For example, a protocol for $RCP_{2.2.1}$ (see Example 1.1.1) is expected to be run in the situation where five cards are given to three agents according to the distribution (2.2.1). It does not make sense to run the protocol in a situation with less agents or more cards. As observed in [FHMV97], it is crucial to make the distinction between the protocol (as the rules governing the actions) and the setting in which it is to be executed. In this chapter, we take epistemic programs as protocols and use a set of logical formulas to specify the *initial assumptions* explicitly. The verification of a protocol w.r.t. the intended goal should then be performed against initial models satisfying such assumptions. Based on the above consideration, we let the protocol specification include not only the protocol and its intended goal but also the initial assumptions:

3.3.1. DEFINITION. (Protocol Specification) A *protocol specification* Prot is a triple $\langle \pi, \phi, T \rangle$ where the protocol π is an epistemic program in L_{AP} , ϕ is a program-free epistemic formula of L_{AP} serving as the intended *goal* of the protocol, and T is a set of program-free epistemic formulas of L_{AP} defining the initial assumptions of the protocol. \mathfrak{M}

A protocol specification Prot is *deterministic* if on any pointed model that satisfies T_{Prot} , there is at most one run of π_{Prot} . It is called *non-deterministic* if it is not deterministic. We also say π_{Prot} is a *deterministic protocol* if Prot is deterministic.

Note that our definition of *determinism* is not based on the syntactic form of π : we allow nondeterministic choices $+$ in a deterministic specification. Let has_jx be the basic proposition meaning: agent j has card x . A simple example of a deterministic protocol specification in a card game setting is:

$$\langle (!_A has_{AC} + !_A has_{Ad}), K_B has_{AC} \vee K_B has_{Ad}, \{(has_{AC} \wedge \neg has_{Ad}) \vee (\neg has_{AC} \wedge has_{Ad})\} \rangle$$

where agent A is to announce his card according to the protocol, in order to let B know his card, under the setting in which A can only have one of the two cards c and d . Note that has_{AC} and has_{Ad} are not logically exclusive, thus can be both true on *some* model. However, such model is excluded by the initial assumption: $(has_{AC} \wedge \neg has_{Ad}) \vee (\neg has_{AC} \wedge has_{Ad})$. Therefore the above specification is deterministic according to our definition.

Intuitively, a (*complete*) *verification* of a protocol specification Prot should check whether the goal ϕ holds after *any* execution of the protocol π_{Prot} on *any* model that satisfies the initial assumptions T_{Prot} . Formally, we need to check:

$$T_{\text{Prot}} \models [\pi_{\text{Prot}}]\phi_{\text{Prot}}$$

If T_{Prot} is a finite set then the above complete verification problem amounts to checking the satisfiability of the L_{AP} -formula $\bigwedge T_{\text{Prot}} \rightarrow [\pi_{\text{Prot}}]\phi_{\text{Prot}}$. However, as we mentioned in the previous section, the satisfiability problem for L_{AP} is undecidable even on finite models. On the other hand, whether complete verification is necessary is actually in question. In practice, we often focus on particular initial models that satisfy T , since T may not be a complete characterization of the intended informal initial requirements we have in mind, namely, there can be unintended models that also satisfy T . The ideal case is that the set T has a unique model, and this model can be generated by a certain method. Such issues related to modelling will be addressed in Part II.

In this chapter, we focus on the verification problem of a protocol specification Prot against a given pointed model (\mathcal{M}, s) that satisfies T_{Prot} . The verification problem then amounts to the following model checking problem:

$$\mathcal{M}, s \models [\pi_{\text{Prot}}]\phi_{\text{Prot}}$$

In some scenarios (e.g., Russian Cards), we are interested in verifying a protocol *universally* in a certain model \mathcal{M} , where each state of \mathcal{M} represents a particular initial distribution of information (e.g., a random card deal). In such cases, the protocol is also required to be executable under arbitrary initial distribution of the information. This *universal verification* of Prot against \mathcal{M} can be formalized as the model checking problem:

$$\mathcal{M} \models (\pi_{\text{Prot}})\phi_{\text{Prot}}$$

namely, for all $s \in S_{\mathcal{M}}$: model checking $\mathcal{M}, s \models \langle \pi_{\text{Prot}} \rangle \phi_{\text{Prot}} \wedge [\pi_{\text{Prot}}]\phi_{\text{Prot}}$.

As we motivated in Chapter 1, the verification of an epistemic protocol is more subtle than it seems to be: the meta-knowledge of the protocol itself may affect the verification of the protocol. We shall see that the above formalizations of the verification problem are not appropriate any more, if we assume the protocol specification is commonly known.

By commonly knowing a *protocol specification* Prot we mean the following:

1. The protocol itself (π_{Prot}) is commonly known;
2. The intended goal of the protocol (ϕ_{Prot}) is commonly known.
3. The set of initial assumptions (T_{Prot}) is commonly known.

If an epistemic protocol is publicly available and is to be used repeatedly, then it is natural to assume the above. Therefore a rigid verification of an epistemic protocol should be undertaken under these meta-knowledge assumptions. Note that the third assumption can be fulfilled by letting the formulas in T be of the form $C\psi$. We will address the issues about the second assumption in Chapter 4. In this chapter we focus on the first assumption and address the verification problem under this assumption.

Let us start from an observation made in [vD03] that checking $\mathcal{M}, s \models [!\pi]\phi$ is sometimes not sufficient, even for a single step protocol $\pi = !\psi$ aiming at establishing ϕ . As we saw in Example 1.1.1, if the agents know the intended goal of the protocol then they will assume that others do not perform actions which can not lead to the goal. The knowledge assumption about the intended goal lets the agents be able to reason more, which may destroy the correctness of the protocol established without the assumption of agents' knowing the goal.

To incorporate this knowledge assumption in the current framework, a straightforward idea is to just announce the intended goal of the protocol thus make it commonly known. In [vD03], the author proposed that, in a Russian Cards setting, the verification of a protocol with the intended goal ϕ should be undertaken while an announcement $!\psi$ is interpreted as more than just announcing ψ ²:

$$\mathcal{M}, w \models [!(\psi \wedge [!\psi]\phi)]\phi$$

The idea is that by announcing ψ as well as the intended effect of the announcement ψ , we may verify the correctness of the protocol under the assumption that agents know the goal. However, if the correctness of $[!(\psi \wedge [!\psi]\phi)]\phi$ is now assumed and known by agents, we still need to make sure that knowing *this* again does not affect the correctness of the protocol. We can iterate such reasoning *ad infinitum*.

Now let us consider an arbitrary protocol π in L_{AP} and a corresponding goal ϕ . We define:

²In the original setting of [vD03], it is suggested that the announcement should be formalized by a *Gricean reading*: the announcement of $!\psi$ by agent a aiming at establishing ψ is formalized as $!(K_a\psi \wedge [!K_a\psi]K_a\phi)$ (the so-called "safe communication"). We omit the details in [vD03] that are relevant to the context of Russian Cards problem.

- $\eta_0 = [\pi]\phi$
- $\eta_{i+1} = [!(\eta_0 \wedge \dots \wedge \eta_i)][\pi]\phi$

We can simplify η_{i+1} by making use of the following valid formula

$$[(\psi \wedge [!\psi]\phi)]\chi \leftrightarrow [!\psi][!\phi]\chi \quad (\star)$$

$$\mathbf{3.3.2. PROPOSITION.} \quad \eta_{i+1} = \underbrace{[!(\pi)\phi] \cdot [!(\pi)\phi] \cdots [!(\pi)\phi]}_i [\pi]\phi$$

PROOF Since

$$\eta_{i+1} = [!(\eta_0 \wedge \dots \wedge \eta_i)]\phi = [!(\eta_0 \wedge \dots \wedge \eta_{i-1} \wedge [!(\eta_0 \wedge \dots \wedge \eta_{i-1})][\pi]\phi)][\pi]\phi.$$

By (\star) , it is not hard to see that

$$\eta_{i+1} = [!(\eta_0 \wedge \dots \wedge \eta_{i-1})][!\pi]\phi.$$

Repeatedly applying the transformation, we have

$$\eta_{i+1} = \underbrace{[!\pi]\phi \cdots [!\pi]\phi}_i [\pi]\phi = \underbrace{[!(\pi)\phi] \cdots [!(\pi)\phi]}_i [\pi]\phi.$$

✱

Intuitively, we need to check η_i for each i , since there are cases where all the η_i are logically different. To see this, recall the Muddy Children scenario 1.1.2, where we showed that if there are n muddy children and it is commonly known that at least one of them is muddy, then the muddy children will only get to know that they are muddy after announcing “we don’t know” for $n-1$ times. Now let π_d be the dummy announcement $!\top$ and let ϕ_n be the formula that expresses “all the n children do not know whether they are muddy or not”. It is clear that $[\pi_d]\phi_n \leftrightarrow \phi_n$ is valid. The above analysis of the Muddy Children scenario tells us that for any $i > 0$, there is always a pointed model (\mathcal{M}_i, s) such that:

$$\mathcal{M}_i, s \models \underbrace{[!(\pi_d)\phi_{i+2}] \cdots [!(\pi_d)\phi_{i+2}]}_i [\pi_d]\phi_{i+2},$$

but

$$\mathcal{M}_i, s \not\models \underbrace{[!(\pi_d)\phi_{i+2}] \cdots [!(\pi_d)\phi_{i+2}]}_{i+1} [\pi_d]\phi_{i+2}$$

It is not hard to see that $[!(\pi)\phi]^*[\pi]\phi$ expresses exactly the infinite conjunction of all the η_i . Thus to verify the protocol π under the assumption of the common

knowledge of the goal of the protocols we need to model check the fixed point formula $[!(\lceil\pi\rceil\phi)^*][\lceil\pi\rceil\phi]$ instead of $[\lceil\pi\rceil\phi]$.

For universal verification, we let $\eta'_0 = \langle\lceil\pi\rceil\phi$ and $\eta'_{i+1} = [!(\eta_0 \wedge \dots \wedge \eta_i)]\langle\lceil\pi\rceil\phi$. Similarly, we can show:

$$\mathbf{3.3.3. PROPOSITION.} \quad \eta'_{i+1} = \underbrace{[!(\langle\lceil\pi\rceil\phi) \cdots !(\langle\lceil\pi\rceil\phi)]}_{i} \langle\lceil\pi\rceil\phi$$

Therefore universal verification under the knowledge assumption of the intended goal amounts to checking $\mathcal{M} \vDash [!(\langle\lceil\pi\rceil\phi)^*]\langle\lceil\pi\rceil\phi$ which can be simplified as follows:

3.3.4. PROPOSITION.

$$\mathcal{M} \vDash [!(\langle\lceil\pi\rceil\phi)^*]\langle\lceil\pi\rceil\phi \iff \mathcal{M} \vDash \langle\lceil\pi\rceil\phi$$

PROOF \Rightarrow is trivial. For \Leftarrow : Suppose $\mathcal{M} \vDash \langle\lceil\pi\rceil\phi$, then announcing $\langle\lceil\pi\rceil\phi$ does not change model \mathcal{M} , thus the truth values of the formulas are preserved after any iteration of the announcement $\langle\lceil\pi\rceil\phi$ ³. Therefore $\mathcal{M} \vDash [!(\langle\lceil\pi\rceil\phi)^*]\langle\lceil\pi\rceil\phi$. \times

We may simplify the verification problem further when looking at *connected* models where every state is connected to all the other states by some path of relations:

3.3.5. PROPOSITION. *If \mathcal{M} is a connected model then for any s in \mathcal{M} ,*

$$\mathcal{M} \vDash [!(\langle\lceil\pi\rceil\phi)^*][\lceil\pi\rceil\phi \iff \mathcal{M}, s \vDash C\langle\lceil\pi\rceil\phi$$

PROOF Immediate from Proposition 3.3.3 and the connectivity. \times

This means that to verify a protocol is correct under any possible initial information distribution is to check the common knowledge of the correctness of the protocol at an *arbitrary* initial situation. Note that in most applications, the initial epistemic model under consideration is indeed connected, assuming that the agents are perfect reasoners who can imagine the possibilities others may think.

In summary, we define the verification of a protocol specification as follows:

3.3.6. DEFINITION. (Verification under Common Knowledge) Given a protocol specification Prot and a pointed model (\mathcal{M}, s) satisfying T_{Prot} , the verification of Prot against (\mathcal{M}, s) under the common knowledge of the intended goal ϕ is checking:

$$\mathcal{M}, s \vDash [!(\lceil\pi_{\text{Prot}}\rceil\phi_{\text{Prot}})^*]\phi_{\text{Prot}}$$

³cf. [vDK06] for a more general study on *successful updates*.

The *universal verification* of Prot against model \mathcal{M}, s under the common knowledge of the intended goal ϕ is checking:

$$\mathcal{M} \models (\pi)\phi_{\text{Prot}}$$

When the model \mathcal{M} is connected, the universal verification problem is equivalent to checking the common knowledge of the correctness of the protocol:

$$\mathcal{M}, s \models C(\pi_{\text{Prot}})\phi_{\text{Prot}}$$

□

Note that since the universal verification problem reduces to simply checking $\mathcal{M} \models (\pi)\phi_{\text{Prot}}$, then given a π without iteration the universal verification can also be done in the standard framework of PAL (see page 17).

3.4 Deterministic Protocols for $RCP_{3,3,1}$

In this section we take the Russian Cards Problem as an example to demonstrate the use of our framework in designing and verifying epistemic protocols. In particular, we study deterministic 2-step protocols for $RCP_{3,3,1}$ that can be executed under an *arbitrary* initial distribution of cards. We show that there is a correct, deterministic protocol for $RCP_{3,3,1}$, however with uneven occurrences of the cards in the announcements (i.e. some cards occur more often than others in the announcements).

Recall that in the setting of $RCP_{n,n,k}$, $2n + k$ cards are distributed among three agents according to the distribution (n,n,k) and the agents can only see their own cards (cf. Example 1.1.1). We first formalize the initial assumptions in this setting.

Let $\mathbf{I} = \{A, B, E\}$ be the set of agents, $Dk_{n,n,k} = \{0, 1, \dots, 2n + k - 1\}$ be the set of $2n + k$ cards, Hs_h be the set of h -hands (e.g., $Hs_3 = \{\{x, y, z\} \mid x, y, z \in Dk_{n,n,k} \text{ are different}\}$). Let us consider a tailored set of basic propositions: $\mathbf{P}_{has} = \{has_i x \mid i \in \mathbf{I}, x \in Dk_{n,n,k}\}$ where $has_i x$ intuitively expresses that agent i has card x . We use $has_i X$ as the abbreviation of $\bigwedge_{x \in X} has_i x$. The initial assumptions are formalized as $T_{n,n,k} = \{C\phi\}$ where:

$$\phi = \text{OneCardInOneP} \wedge \text{EkCards} \wedge \text{ABnCards} \wedge \text{KnowThyself} \wedge \text{DontKnowOthers}$$

- $\text{EkCards} := \bigvee_{x \in Hs_k} has_E x$;
- $\text{ABnCards} := \bigwedge_{i \in \{A, B\}} \bigvee_{x \in Hs_n} has_i x$;
- $\text{OneCardInOneP} := \bigwedge_{i \neq j} (\bigwedge_{x \in Dk_{n,n,k}} (has_i x \rightarrow \neg has_j x))$;
- $\text{KnowThyself} := \bigwedge_{i \in \mathbf{I}} \bigwedge_{x \in Dk_{n,n,k}} (has_i x \rightarrow K_i has_i x)$;
- $\text{DontKnowOthers} := \bigwedge_{i \neq j} \bigwedge_{x \in Dk_{n,n,k}} (has_i x \rightarrow (\widehat{K}_j has_i x \wedge \widehat{K}_j \neg has_i x))^4$.

⁴Recall that $\widehat{K}\phi$ denotes $\neg K_i \neg \phi$ (i thinks that ϕ is possible).

Intuitively, the formula in T says that it is commonly known that $2n + k$ cards are distributed among three agents according to the distribution $(n.n.k)$ and the agents only know their own cards.

Now we build an initial model which satisfies the above initial assumptions. Let $\mathcal{M}_{n.n.k} = \{S, \mathbf{P}_{has}, \mathbf{I}, \sim, V\}$ where:

- $S = \{(X, Y, Z) \mid X, Y \in Hs_n, Z \in Hs_k, X \cup Y \cup Z = Dk_{n.n.k}\}$;
- $s \sim_i t \iff s_i = t_i$ where $(X, Y, Z)_A = X$, $(X, Y, Z)_B = Y$ and $(X, Y, Z)_E = Z$;
- $V(s) = \{has_i x \mid i \in \mathbf{I}, x \in Dk_{n.n.k} \text{ and } x \text{ appears in } s_i\}$.

Note that $\mathcal{M}_{n.n.k}$ is clearly connected. It can be verified that $\mathcal{M}_{n.n.k}$ satisfies $T_{n.n.k}$.

A correct protocol for $RCP_{n.n.k}$ should let A and B eventually know each other's cards (thus also E 's cards) while keeping E ignorant about A 's and B 's cards. This can be formalized as $\phi_{n.n.k} = \phi_1 \wedge \phi_2 \wedge \phi_3$ where:

$$\begin{aligned}\phi_1 &= \bigwedge_{x \in Dk_{n.n.k}} (has_{Ax} \rightarrow K_B has_{Ax}) \\ \phi_2 &= \bigwedge_{x \in Dk_{n.n.k}} (has_{Bx} \rightarrow K_A has_{Bx}) \\ \phi_3 &= \bigwedge_{x \in Dk_{n.n.k}} ((has_{Ax} \rightarrow \neg K_E has_{Ax}) \wedge (has_{Bx} \rightarrow \neg K_E has_{Bx}))\end{aligned}$$

Recall that in Example 1.1.1 we argue that knowing the correctness of the protocol may destroy the correctness of the protocol. Now we can make this claim formal: let $\pi = !(has_{A01} \vee has_{A23} \vee has_{A24} \vee has_{A34})$, it is easy to check that

$$\mathcal{M}_{2.2.1}, (01, 23, 4) \models [\pi](\phi_1 \wedge \neg K_E has_{A01})$$

but

$$\mathcal{M}_{2.2.1}, (01, 23, 4) \models [![\pi]\phi_1]K_E has_{A01}$$

In the rest of this section, we focus on finding a protocol $\pi_{3.3.1}$ such that the following conditions are met:

1. $\pi_{3.3.1}$ is a two-step protocol in the form $\pi_1 \cdot \pi_2$ where $\pi_1 = !_A \psi_1 + \dots + !_A \psi_m$ and $!_B \psi'_1 + \dots + !_B \psi'_l$ for some ψ_i and ψ'_i .
2. $\langle \pi_{3.3.1}, \phi_{3.3.1}, T_{3.3.1} \rangle$ is deterministic;
3. $\mathcal{M}_{3.3.1} \models \langle \pi_{3.3.1} \rangle (\phi_1 \wedge \phi_2 \wedge \phi_3)$;

We say a deterministic, 2-step protocol is *correct* for $RCP_{3.3.1}$ if condition (3) is met. Note that a protocol satisfying (2) can have at most one run at each of the states in $\mathcal{M}_{3.3.1}$. Therefore, assuming (2), checking (3) is then equivalent to checking:

$$\mathcal{M}_{3.3.1} \models \langle \pi_{3.3.1} \rangle (\phi_1 \wedge \phi_2 \wedge \phi_3)$$

The following proposition suggests that we can focus on the first step of the protocol with an intermediate goal $\phi_1 \wedge \phi_3$.

3.4.1. PROPOSITION. *If there is a one-step protocol π_1 such that $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle (\phi_1 \wedge \phi_3)$, then there is a π_2 such that $\mathcal{M}_{n,n,k} \models \langle \pi_1 \cdot \pi_2 \rangle (\phi_1 \wedge \phi_2 \wedge \phi_3)$.*

PROOF Let $\pi_2 = \sum_{X \in Hs_k} !_B has_E X$, we show that if $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle (\phi_1 \wedge \phi_3)$ then $\mathcal{M}_{n,n,k} \models \langle \pi_1 \cdot \pi_2 \rangle (\phi_1 \wedge \phi_2 \wedge \phi_3)$. Let $K_i Cards_j$ be the abbreviation for $\bigwedge_{x \in Dk} (has_j x \rightarrow K_i has_j x)$, then $\phi_1 = K_B Cards_A$ and $\phi_2 = K_A Cards_B$. We can verify that if $i, j, l \in \mathbf{I}$ are different then $K_i Cards_j \rightarrow K_i Cards_l(\phi_2)$ is valid in any submodel of $\mathcal{M}_{n,n,k}$. Therefore it is easy to see that if $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle \phi_1$ then $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle K_B Cards_E$. Thus $\pi_1 \cdot \pi_2$ is executable and $\mathcal{M}_{n,n,k} \models \langle \pi_1 \cdot \pi_2 \rangle K_A Cards_E$. It follows that $\mathcal{M}_{n,n,k} \models \langle \pi_1 \cdot \pi_2 \rangle \phi_2$. Since ϕ_1 is clearly preserved after executing π_2 , we only need to show $\mathcal{M}_{n,n,k} \models \langle \pi_1 \cdot \pi_2 \rangle \phi_3$. We claim that

$$\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle C(K_B Cards_E)$$

If the claim is true, then $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle K_E(K_B Cards_E)$ thus for any state $s = (X, Y, Z) : \mathcal{M}_{n,n,k,r} s \models \langle \pi_1 \rangle K_E(K_B has_E Z)$. Therefore truthfully announcing $K_B has_E Z$ for some $Z \in Hs_k$ does not change the distributions of hands that E considers possible. Therefore ϕ_3 should be preserved after π_2 .

Now we prove the claim. First note that π_2 is clearly deterministic: it can have only one run when executed. Now given an arbitrary s in $\mathcal{M}_{n,n,k,r}$ if $s \sim_{\mathbf{I}} t$ in a model \mathcal{M}' such that $(\mathcal{M}, s) \Vdash \langle \pi_2 \rangle (\mathcal{M}', s)$ then we know there is a unique run w such that $w \in \mathcal{L}(\pi_2)$ and $(\mathcal{M}_{n,n,k,r} s) \Vdash [w] (\mathcal{M}', s)$ and $(\mathcal{M}_{n,n,k,r} t) \Vdash [w] (\mathcal{M}', t)$. Since $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle K_B Cards_E$ then $\mathcal{M}', t \models K_B Cards_E$. Therefore $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle C(K_B Cards_E)$ and this proves the claim. \spadesuit

To simplify the discussion, we can restrict the form of π_1 further w.l.o.g by adapting a result from [vD03], which states that to announce only A 's alternative hands is enough.

3.4.2. PROPOSITION. *If $\mathcal{M}_{n,n,k} \models \langle \pi_1 \rangle (\phi_1 \wedge \phi_3)$ then there is a π'_1 such that $\mathcal{M}_{n,n,k} \models \langle \pi'_1 \rangle (\phi_1 \wedge \phi_3)$ and*

$$\pi'_1 = !_A Pa_0 + !_A Pa_1 + \dots + !_A Pa_m$$

where Pa_i is of the form: $has_A X_0 \vee has_A X_1 \vee \dots \vee has_A X_i$ with $\{X_0, \dots, X_i\} \subseteq Hs_n$.

PROOF The crucial observation for the proof is that for any formula ϕ such that $\mathcal{M}_{n,n,k,r} s \models K_A \phi$ at some s , $\{t \mid \mathcal{M}_{n,n,k,r} t \models \phi\}$ must be the union of some i -equivalence classes in $\mathcal{M}_{n,n,k}$ and each equivalence class of $\mathcal{M}_{n,n,k}$ can be characterized by a formula $has_A X$ for some hand $X \in Hs_n$. Therefore, for each ϕ such that $\mathcal{M}_{n,n,k,r} s \models K_A \phi$ at some s there is a $\phi' = has_A X_0 \vee has_A X_1 \vee \dots \vee has_A X_m$ for a set of hands $\{X_0, \dots, X_m\} \subseteq Hs_n$ such that $\{t \mid \mathcal{M}_{n,n,k,r} t \models \phi\} = \{t \mid \mathcal{M}_{n,n,k,r} t \models \phi'\}$. \spadesuit

In the sequel, we sometimes abuse the notion of Pa_i by viewing it as a set of 3-hands with each hand represented by xyz for $\{x, y, z\} \subset Dk_{n,n,k}$ (the order does not matter). We now prove a lemma for later use in this section.

3.4.3. LEMMA. *If there is a deterministic protocol π_1 such that $\mathcal{M}_{n,n,k} \models (\pi_1)(\phi_1 \wedge \phi_3)$ and $\pi_1 = !_A Pa_0 + !_A Pa_1 + \dots + !_A Pa_m$ then*

1. *each possible hand appears only once in $Pa_0, !Pa_1, \dots, !Pa_m$: for each $X \in Hs_n$: there is a unique $i \leq m$ such that $X \in Pa_i$.*
2. *any two hands in one announcement Pa_j can only share at most $n - k - 1$ cards.*

PROOF For (1): Since $\mathcal{M}_{n,n,k} \models (\pi_1)\top$ and $\pi_1 = !_A Pa_0 + !_A Pa_1 + \dots + !_A Pa_m$ we know that every hand should appear at least once. From the assumption that protocol π is deterministic, every hand can only appear once. In the following, given a hand X of A , let $Pa(X)$ be the set Pa_j such that $X \in Pa_j$.

For (2): To let B know A 's cards after A 's announcement, we should make sure that given A 's hand X , for any B 's hand Z , the alternatives in $Pa(X)$ will be ruled out. Namely, for any different hands $X, Y \in Pa(X)$, any hand $Z \subseteq Dk_{n,n,k} \setminus X$ that B may have: $Z \cap Y \neq \emptyset$. This means that for every two hands X, Y in Pa_i , the number of cards different from the cards in $X \cup Y$ must be less than n . Otherwise there is a possible hand Z which does not intersect with both X and Y . Thus, we have $|Dk_{n,n,k}| - |Y \cup X| < n$. Since $|Dk_{n,n,k}| = 2n+k, |Y \cup X| > n+k$. Note that $|X| = |Y| = n$, thus $|X \cap Y| < 2n - n - k = n - k$. \spadesuit

In the following we will concentrate on the specific case $RCP_{3,3,1}$. We first show that there is a deterministic protocol:

3.4.4. THEOREM. *There is a correct, 2-step deterministic protocol for $RCP_{3,3,1}$ ⁵.*

PROOF Let Pa_i be the following sets of 3-hands.

Pa_0 :	012	036	045	134	156	235	246
Pa_1 :	013	025	046	126	145	234	356
Pa_2 :	014	026	035	136	245		
Pa_3 :	015	024	123	256	346		
Pa_4 :	016	034	124	135	236	456	
Pa_5 :	023	056	125	146	345		

Let $\pi_1 = \sum_{0 \leq i \leq 5} (!_A Pa_i)$. Note that π_1 is clearly a deterministic protocol: if $has_A 025$ is true then A should announce Pa_2 ⁶:

$$has_A 013 \vee has_A 025 \vee has_A 046 \vee has_A 126 \vee has_A 145 \vee has_A 234 \vee has_A 356$$

Moreover we can verify that $\mathcal{M}_{3,3,1} \models (\pi_1)(\phi_1 \wedge \phi_3)$. Thus from Proposition 3.4.1, there is a deterministic 2-step protocol for $RCP_{3,3,1}$. \spadesuit

However, the above protocol is *biased* in the sense that not all the cards appear evenly in each announcement Pa_i (e.g. in Pa_2 , 0 appears three times but others only

⁵The solution was found with the help of the Alloy Analyzer [Jac02] based on Lemma 3.4.3, see Appendix A for the code.

⁶We omit the K_A in front of each $has_A X$.

appear twice). Thus E may learn that some card is more likely than other cards to be held by A . The authors of [AvDR09] showed that we can resolve the harm of *unbiased* protocols by making use of probabilistic selections of announcements. Here we are interested in whether we can have an unbiased deterministic protocol with the same number of occurrences of cards in the alternative announcements. Here are some properties of the unbiased protocol for $RCP_{3,3,1}$, if it exists:

3.4.5. LEMMA. *The first step of an unbiased deterministic protocol for $RCP_{3,3,1}$ must satisfy the following:*

1. *each announcement Pa_i must be a set of 7 alternative hands.*
2. *there are in total 5 alternative announcements in the protocol: Pa_0, \dots, Pa_4 .*
3. *every two hands in the same announcement Pa_i have exactly one card in common.*

PROOF For (1): Suppose all the cards appear evenly (suppose g times) in an announcement with h hands. Since each hand has three cards then $3h = 7g$. So the minimal h is 7, and each card appears 3 times. We claim that if h is greater than 7 then there must be two hands which share more than 1 card. Note that there are only $C_7^2 = 21$ different pairs of cards and the three cards in each hand can constitute 3 different pairs. From the second statement in Lemma 3.4.3 any two hands should not have a pair of cards in common, thus 7 hands of three cards then “cover” all the possible 21 pairs of cards without repetition. Therefore adding one more hand of three cards must result in two hands sharing two cards in common.

For (2): From the first statement in Lemma 3.4.3, we know the $C_7^3 = 35$ hands should all appear in some Pa_i once. Thus from (1) the protocol should have 5 alternative announcements with 7 hands each.

For (3): Suppose there are two hands X, Y in an announcement Pa_j such that $X \cap Y = \emptyset$. Without loss of generality let $X = 123$ and $Y = 456$. Since each of the possible 21 pairs should appear in some hand in the announcement Pa_i as argued in (1), then the hands $14c$ and $24c'$ must also appear in Pa_j for some cards c, c' . Since every two hands should not have two cards in common and $1, 2 \in X$ and $4 \in Y, c, c' \notin X \cup Y$ thus $c = c' = 0$. However then $14c$ and $24c'$ have two cards in common, contradiction. \times

In the following, we show that there is no deterministic protocol which is unbiased.

3.4.6. THEOREM. *There is no correct deterministic 2-step protocol which is unbiased for $RCP_{3,3,1}$.*

PROOF We prove the theorem by proving the following stronger claim first:

There are no 3 sets with 7 hands, such that: (1) all the 21 hands that appear in these sets are different; (2) every two hands in the same set have one and only one common card; (3) all the cards appear evenly in every set.

Suppose towards a contradiction that there exist 3 sets Pa_2, Pa_3, Pa_4 satisfying (1), (2) and (3). Assume without loss of generality that $012 \in Pa_2, 013 \in Pa_3$ and $014 \in Pa_4$.

Since for any $x \in \{2, 3, 4\}$: $01x \in Pa_x$, then from (2) and (3) we know that $xab, xcd \in Pa_x$, $0ac, 0bd \in Pa_x$ and $1ad, 1bc \in Pa_x$ such that $a, b, c, d \in Dk_{3.3.1} \setminus \{0, 1, x\}$ are all different. Now we can list all the hands in Pa_x (with $p_i^x \in \{0, 1, x\}$ and $p_i^x \neq p_j^x$ if $i \neq j$):

$$\begin{array}{llll} \text{for } Pa_2 : 012 & p_1^2 34, p_1^2 56 & p_2^2 35, p_2^2 46 & p_3^2 36, p_3^2 45 \\ \text{for } Pa_3 : 013 & p_1^3 24, p_1^3 56 & p_2^3 25, p_2^3 46 & p_3^3 26, p_3^3 45 \\ \text{for } Pa_4 : 014 & p_1^4 23, p_1^4 56 & p_2^4 25, p_2^4 36 & p_3^4 26, p_3^4 35 \end{array}$$

First, there exists an $x \in \{2, 3, 4\}$ such that $p_i^x = x$, otherwise there must be either two 056 or two 156 in Pa_2, Pa_3, Pa_4 , contradictory to (1). Suppose w.l.o.g. that $p_1^2 = 2$. It is easy to see that $p_1^3 \neq 3$ and $p_1^4 \neq 4$, otherwise 234 appears twice in Pa_2, Pa_3, Pa_4 . Moreover, since $p_1^3 56$ is in Pa_3 and $p_1^4 56$ is in Pa_4 , $p_1^3 \neq p_1^4$. Suppose w.l.g. that $p_2^3 = 3$. Then $p_2^4 \neq 4$ since $346 \in Pa_3$, therefore $p_3^4 = 4$. Now let us fill in the known p_i^x as following:

$$\begin{array}{llll} \text{for } Pa_2 : 012 & 234, 256 & p_2^2 35, p_2^2 46 & p_3^2 36, p_3^2 45 \\ \text{for } Pa_3 : 013 & p_1^3 24, p_1^3 56 & 325, 346 & p_3^3 26, p_3^3 45 \\ \text{for } Pa_4 : 014 & p_1^4 23, p_1^4 56 & p_2^4 25, p_2^4 36 & 426, 435 \end{array}$$

Now we have $p_2^2, p_3^2, p_1^3, p_3^3, p_1^4, p_2^4 \in \{0, 1\}$. We showed that $p_1^3 \neq p_1^4$, thus $p_3^3 \neq p_3^4$ (remember that $p_3^3 \neq p_1^3$ and $p_1^4 \neq p_2^4$). Since $p_2^3, p_3^3, p_2^4 \in \{0, 1\}$ then from $p_3^3 \neq p_3^4$ we have: $p_2^3 = p_3^3$ or $p_2^3 = p_2^4$, but in any case, there will be one hand that appears in two announcements, contradiction.

The Theorem follows from the above claim and Lemma 3.4.5. \star

3.5 Conclusion and Discussion

The logical framework developed in this chapter made it possible to formally specify and verify announcement protocols under the assumption that the intended goals of the protocols are commonly known. More examples of the protocol verification using this framework can be found in [WKvE09]. In this chapter, we have restricted ourselves to protocols involving public announcement only. This restriction gives us a straightforward model checking algorithm. As we mentioned before, the Kleene star over announcements is the source of the undecidability, thus it is important to understand the behaviour of the Kleene star better. It is shown in [GK03] that the *limit* of an iterated announcement corresponds to a *deflationary* fixed point on a non-monotonic function, thus can be expressed by a formula in the *Modal Iteration Calculus* (MIC). However, it is not clear whether our logic can be translated back to MIC. Moreover, how to fit the iteration of more general action models in a fixed point logic is also open. Some moderate extensions of the language are subgroup announcements and actions for factual change (cf. [vDK08, vBvEK06]). Another interesting extension is to add concurrent actions for modelling simultaneous announcements in a distributed setting which will be addressed in Chapter 5. The restriction to

public announcements also gives some hope for the synthesis problem of epistemic protocols (cf. the ideas mentioned [ABvDS09]).

Another important restriction in the current framework is that we exclude explicit tests in the program language of L_{AP} (we do have an intrinsic test $?\phi$ implicitly for each announcement $!\phi$). Therefore we can not specify guarded announcements like $?K_i(p \wedge q) \cdot !Kp$, where the test is not the announcement itself. A straightforward idea is to introduce the tests with its usual semantics as in PDL. However, as we shall see in Chapter 4, explicit tests require more careful modifications of the semantics, due to the fact that non-intrinsic tests are not publicly observable. As we remarked in the introduction, the tests can explicitly model the preconditions of the actions which make the actions carry more meaning than they seem to have. The link between the precondition and the actions should be established by protocols known to the agents. In Chapter 4, we will have an extensive discussion on how to know and change the protocols.